MATH 282 Analysis of Algorithm’s Complexity

**Algorithm:** Sequential search

**Factor to be analyzed:** Time (number of steps, speed) / Space

**Situation to be analyzed:** Worst case / Average case / Best case / OTHER (special situation)

**Explanation of situation:** Item to be searched for is not in the list or item is last in list

**Key step (reflects work done):** Comparison

**Parameter for analysis:** *n* is the number of items in the list

*(what determines how the algorithm’s efficiency changes as the size of data increases?)*

**Questions/Process:**

* Are there any steps which are not simple steps (comparable to the key step)?
  + If so, what is the complexity of those steps (relative to the key step)? Factor into the analysis.
* If desired, count the number of times each step is carried out (or just the key step).
* Identify each loop and determine how many times the loop is carried out (in relation to *n*).
* How are the loops related?
  + If nested, multiply the steps.
  + If separate, add the steps.
* Eliminate any constants and any lower-level terms.

Let A represent an assignment operation, C represent a comparison, B represent an array access, and D represent an addition. Steps A, B, C, and D are all simple steps, similar in workload to a comparison. Then there are the following steps:

Focus on comparisons: there are comparisons.

Eliminate the constants and the lower-level terms: The order of the algorithm in the worst case is *n*.

For(int i=0;i<arr.length;i++) if arr[i]==key break;

Alternative analysis: There is one loop, and in the worst case, the loop is repeated *n* times (once for every item in the list).

**Result:** Algorithm is O( *n* )